

**Московский государственный технический
университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по рубежному контролю № 2
Вариант Г-12

Выполнил:
студент группы ИУ5-34Б
Сафронов Андрей

Проверил:
преподаватель каф. ИУ5
Нардид А. Н.

Москва, 2024 г.

1) Был проведен рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования:

```
class ProgrammingLanguage:
    """Язык программирования"""

    def __init__(self, id, name, ide_id):
        self.id = id
        self.name = name
        self.ide_id = ide_id

class IDE:
    """Средство разработки (IDE)"""

    def __init__(self, id, name):
        self.id = id
        self.name = name

class LanguageIDE:
    """Языки программирования в средах разработки' для реализации связи многие-ко-многим"""

    def __init__(self, ide_id, language_id, users_count):
        self.ide_id = ide_id
        self.language_id = language_id
        self.users_count = users_count

def create_one_to_many(ides, languages):
    """Соединение данных один-ко-многим"""
    return [(lang.name, ide.id, ide.name)
            for ide in ides
            for lang in languages
            if lang.ide_id == ide.id]

def create_many_to_many(languages_ides, languages, ides):
    """Соединение данных многие-ко-многим"""
    return [(lang.name, ide.name, lid.users_count)
            for ide in ides
            for lid in languages_ides
            for lang in languages if lang.id == lid.language_id and ide.id == lid.ide_id]

def task_1(one_to_many, string):
    return list(filter(lambda x: x[2].startswith(string), one_to_many))

def task_2(many_to_many):
    max_users = {}
```

```

for lang_name, ide_name, users_count in many_to_many:
    if ide_name not in max_users or max_users[ide_name] < users_count:
        max_users[ide_name] = users_count
return sorted(max_users.items(), key=lambda x: x[1])

def task_3(many_to_many):
    return sorted(many_to_many, key=lambda x: (x[1], x[0]))

def main():
    # Список средств разработки (IDE)
    ides = [
        IDE(1, 'PyCharm'),
        IDE(2, 'Visual Studio Code'),
        IDE(3, 'IntelliJ IDEA'),
        IDE(4, 'Eclipse'),
        IDE(5, 'Sublime Text'),
        IDE(6, 'Visual Studio')
    ]

    # Список языков программирования
    languages = [
        ProgrammingLanguage(1, 'Python', 1),
        ProgrammingLanguage(2, 'Java', 3),
        ProgrammingLanguage(3, 'C++', 4),
        ProgrammingLanguage(4, 'JavaScript', 2),
        ProgrammingLanguage(5, 'C#', 6),
        ProgrammingLanguage(6, 'Go', 2)
    ]

    # Связь языков программирования и средств разработки
    languages_ides = [
        LanguageIDE(1, 1, 1000),
        LanguageIDE(2, 4, 2000),
        LanguageIDE(3, 2, 1500),
        LanguageIDE(4, 3, 1600),
        LanguageIDE(2, 6, 200),
        LanguageIDE(5, 5, 800),
        LanguageIDE(6, 5, 1200),
        LanguageIDE(5, 1, 200),
        LanguageIDE(4, 6, 700),
        LanguageIDE(1, 4, 500),
        LanguageIDE(3, 5, 300),
        LanguageIDE(2, 2, 400),
        LanguageIDE(6, 3, 600),
    ]

    one_to_many = create_one_to_many(ides, languages)
    many_to_many = create_many_to_many(languages_ides, languages, ides)

    # Г1: Выводим список всех IDE, которые начинаются с буквы «V»
    print("Задание Г1")

```

```

result = task_1(one_to_many, 'V')
for i in result:
    print(f'Средство разработки: {i[2]}, Язык программирования: {i[0]}")

# Г2: Список средств разработки с максимальным количеством пользователей для
каждого языка
print("\nЗадание Г2")
sorted_ids = task_2(many_to_many)
for ide_name, users_count in sorted_ids:
    print(f'Средство разработки: {ide_name}, Максимальное количество пользователей:
{users_count}')

# Г3: Список всех связанных языков программирования и средств разработки
print("\nЗадание Г3")
sorted_languages = task_3(many_to_many)
for lang in sorted_languages:
    print(f'Средство разработки: {lang[1]}, Язык программирования: {lang[0]}")

if __name__ == '__main__':
    main()

```

2) Для текста программы рубежного контроля №1 были созданы модульные тесты с применением TDD - фреймворка (3 теста):

```

import main
import unittest

class TestMethods(unittest.TestCase):

    def setUp(self):
        self.ides = [
            main.IDE(1, 'PyCharm'),
            main.IDE(2, 'Visual Studio Code'),
            main.IDE(3, 'IntelliJ IDEA'),
            main.IDE(4, 'Eclipse'),
            main.IDE(5, 'Sublime Text'),
            main.IDE(6, 'Visual Studio')
        ]
        self.languages = [
            main.ProgrammingLanguage(1, 'Python', 1),
            main.ProgrammingLanguage(2, 'Java', 3),
            main.ProgrammingLanguage(3, 'C++', 4),
            main.ProgrammingLanguage(4, 'JavaScript', 2),
            main.ProgrammingLanguage(5, 'C#', 6),
            main.ProgrammingLanguage(6, 'Go', 2)
        ]
        self.languages_ids = [
            main.LanguageIDE(1, 1, 1000),
            main.LanguageIDE(2, 4, 2000),
            main.LanguageIDE(3, 2, 1500),
            main.LanguageIDE(4, 3, 1600),

```

```

        main.LanguageIDE(2, 6, 200),
        main.LanguageIDE(5, 5, 800),
        main.LanguageIDE(6, 5, 1200),
        main.LanguageIDE(5, 1, 200),
        main.LanguageIDE(4, 6, 700),
        main.LanguageIDE(1, 4, 500),
        main.LanguageIDE(3, 5, 300),
        main.LanguageIDE(2, 2, 400),
        main.LanguageIDE(6, 3, 600),
    ]

    self.one_to_many = main.create_one_to_many(self.ides, self.languages)
    self.many_to_many = main.create_many_to_many(self.languages_ides,
self.languages, self.ides)

    def test_first_task_method(self):
        result = [(i[2],i[0]) for i in main.task_1(self.one_to_many, 'V')]
        true_result = [('Visual Studio Code', 'JavaScript'),
                        ('Visual Studio Code', 'Go'),
                        ('Visual Studio', 'C#')]
        self.assertEqual(result, true_result)

    def test_second_task_method(self):
        result = main.task_2(self.many_to_many)
        true_result = [('Sublime Text', 800),
                        ('PyCharm', 1000),
                        ('Visual Studio', 1200),
                        ('IntelliJ IDEA', 1500),
                        ('Eclipse', 1600),
                        ('Visual Studio Code', 2000)]
        self.assertEqual(result, true_result)

    def test_third_task_method(self):
        result = [(i[1], i[0]) for i in main.task_3(self.many_to_many)]
        true_result = [('Eclipse', 'C++'),
                        ('Eclipse', 'Go'),
                        ('IntelliJ IDEA', 'C#'),
                        ('IntelliJ IDEA', 'Java'),
                        ('PyCharm', 'JavaScript'),
                        ('PyCharm', 'Python'),
                        ('Sublime Text', 'C#'),
                        ('Sublime Text', 'Python'),
                        ('Visual Studio', 'C#'),
                        ('Visual Studio', 'C++'),
                        ('Visual Studio Code', 'Go'),
                        ('Visual Studio Code', 'Java'),
                        ('Visual Studio Code', 'JavaScript')]
        self.assertEqual(result, true_result)

if __name__ == '__main__':
    unittest.main()

```

Результат выполнения программы:

```
Testing started at 11:08 ...  
Launching unittests with arguments python -m unittest E:\GitHub\labs_3_sem\Python\RK2\unit_test.py in E:\GitHub\labs_3_sem\Python\RK2  
  
Ran 3 tests in 0.001s  
  
OK  
  
Process finished with exit code 0
```